DTU Library

Geometric data transfer between CAD systems: solid models

Kroszynski, Uri; Palstroem, Bjarne; Trostmann, Erik; Schlechtendahl, E. G.

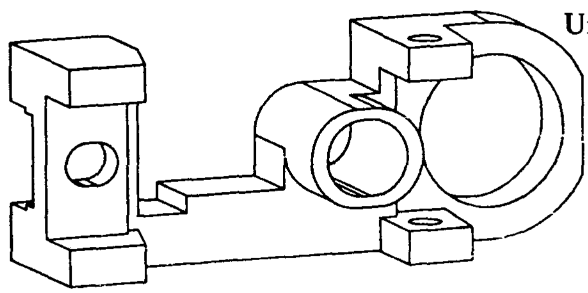
Published in:
I E E E Computer Graphics and Applications

Link to article, DOI:
10.1109/38.35538

Publication date:
1989

Document Version
Publisher's PDF, also known as Version of record

Geometric Data Transfer Between CAD Systems: Solid Models



Uri I. Kroszynski, Bjarne Palstroem, and Erik Trostmann
Technical University of Denmark

Ernst G. Schlechtendahl
Kernforschungszentrum Karlsruhe

The CAD/CAM community is showing great interest in the development of data exchange standards. This article highlights some aspects of the work accomplished during the first phase of the ESPRIT project CAD*I. Part of this work has resulted in a specification for the exchange of solid models, as well as in some pilot implementations of processors based on this specification. This work also contributes to the efforts of the International Standards Organization (ISO) in developing the Standard for the Exchange of Product Model Data (STEP).

We present a summary of the CAD*I approach, addressing the structure of neutral files for solids, entities, and attributes supporting three kinds of representations; facilities for the transfer of parametric designs; referencing library components; and other general mechanisms.

We also describe the current state of the specification and processor implementations, and include an example of a CAD*I neutral file. Results from cycle and intersystem solid model transfer tests are presented, showing the practicality of the CAD*I proposal. B-rep model transfer results are discussed in some detail. Finally, we outline the relationship of this work with standardization.

The continuous and rapid growth of the CAD/CAM community has intensified the need for communicating data between different CAD systems as well as from CAD to such CIM areas as analysis, manufacturing, and quality assurance. The diversity of CAD equipment and installations, the ever-increasing complexity of products modeled, and the multitude of individual computer-aided applications make fully realizing the potentials of integrated design, analysis, and production very difficult.

Leaving aside the topics associated with electronic data transmission, we focus on the translation of digital representations of geometric models and associated information in CAD environments to and from neutral representations, which we call CAD interfaces. In particular, we concentrate on a proposal for solids developed by CAD*I.

Numerous attempts to provide flexible, yet stable methods of product data transfer have resulted in quite a few practical interfaces. Several protocols for the transfer of CAD data are currently being employed with more or less success. The best known internationally is the Initial Graphics Exchange Specification (IGES) 1.0, which became the central part of the ANSI Y14.26M standard.[1] As the name suggests, IGES was originally meant for the transfer of drawing data, although it eventually evolved to handle more general product information. In practice, IGES 2.0, 3.0, or 4.0 is used as the reference for most commercial processors. Other protocols—for example, VDAFS in Germany[2] and SET in France[3]—have also become national standards successful in the range of their applicability.

Proposals to introduce solid model descriptions complying with IGES have been developed[4,5] for the most popular representation methods, namely constructive solid geometry (CSG) and boundary representation (B-rep). The major ideas of these efforts were incorporated in Chapter 5 of the ANSI standard.[1] CSG solid representations are included in IGES 4.0, and incorporation of B-reps in Version 5.0 is planned. Wilson has traced the history of these developments.[6]

A procedural interface for solid model transfer, the Application Interface Specification (AIS), as well as

0272-17-16/89/0900-0057$01.00 © 1989 IEEE

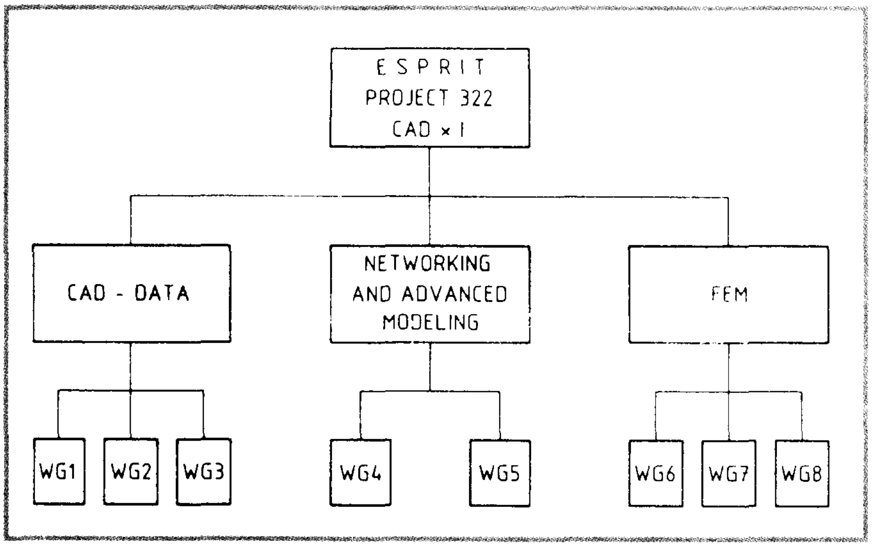


Figure 1. CAD*I project overview.

translators between AIS and the Experimental Boundary File of CAM-I,[5] were published only a few years ago.[7] The Product Data Definition Interface (PDDI)[8] includes B-reps. A new version of the French SET is currently under development. Other ongoing projects in this area are the Product Data Exchange Specification (PDES)[9] and the European Strategic Program for Research in Information Technology, ESPRIT Project 322, CAD Interfaces (CAD*I).[10]

CAD*I was initiated in 1984. The project aims at rationalizing the European efforts to search for unified tools and methods of transfer of product information. CAD data transfer was recognized as one of the key research and development areas of the project for two reasons: It is vitally important in Europe's typically interleaved structure of small and large companies, often in different countries, which participate in industrial product design. Also, current standards are limited in capabilities or efficiency.

Among the eight task groups, Working Group 2 (WG2) deals with the transfer of solid models (see Figure 1). This group is composed of members from research institutions and engineering firms in Denmark, West Germany, Great Britain, and France. Some of the goals of WG2 are to specify a neutral file format for the transfer of solid models, and to develop preprocessors and postprocessors for a number of representative commercial CAD solid modeling systems. A further goal of the CAD*I project as a whole is to contribute to standardization activities nationally as well as internationally via ISO.

An important early task in WG2 was to select and define a language and the structure of the neutral file, so as to be able to express attributes, properties, and entities, as well as the associations among them. These relations are, by the nature of CAD data, partly hierarchical and partly of the networklike "many-to-many" type. Rules for the structure and syntax were worked out and adopted by all working groups. Complying with those rules, each working group defined the keywords and semantics for the entities and associations particular to its area of application, in a so-called reference scheme.

Some properties of the CAD*I neutral file structure and language will be outlined in this article, following the scheme for solid models, as released in Version 2.1 of the CAD*I specification.[11] A section with the results of solid model transfer via CAD*I neutral files in cycle and intersystem tests illustrates the feasibility of the approach.

The experience gained from these tests, as well as discussions within CAD*I and the ISO technical community, have led to some modifications and general upgrades. These, together with a comprehensive set of wireframe and surface model entities, have been incorporated in Versions 3.2 and 3.3,[12,13] which merge the contributions of Working Groups 1, 2, and 3. These versions correspond to a reference scheme for CAD geometry data, rather than being restricted to solids. For geometry a top-down approach was selected, so wireframes become a subset of the entities defined for solids. The merging demonstrates the viability of this approach.

The different representations of solids

The widespread use of solid modeling in CAD for mechanical engineering has brought maturity and stability to the diverse techniques and algorithms. Rather than converging to a unified method, many internal digital representation methods exist, each one with its pros and cons. Therefore, the neutral medium has to be general enough to accommodate many different representations. Reading information from a neutral format file into a particular CAD system (postprocessing), implies sorting out the portion of the file that the system can understand and trying to resolve the rest in the best

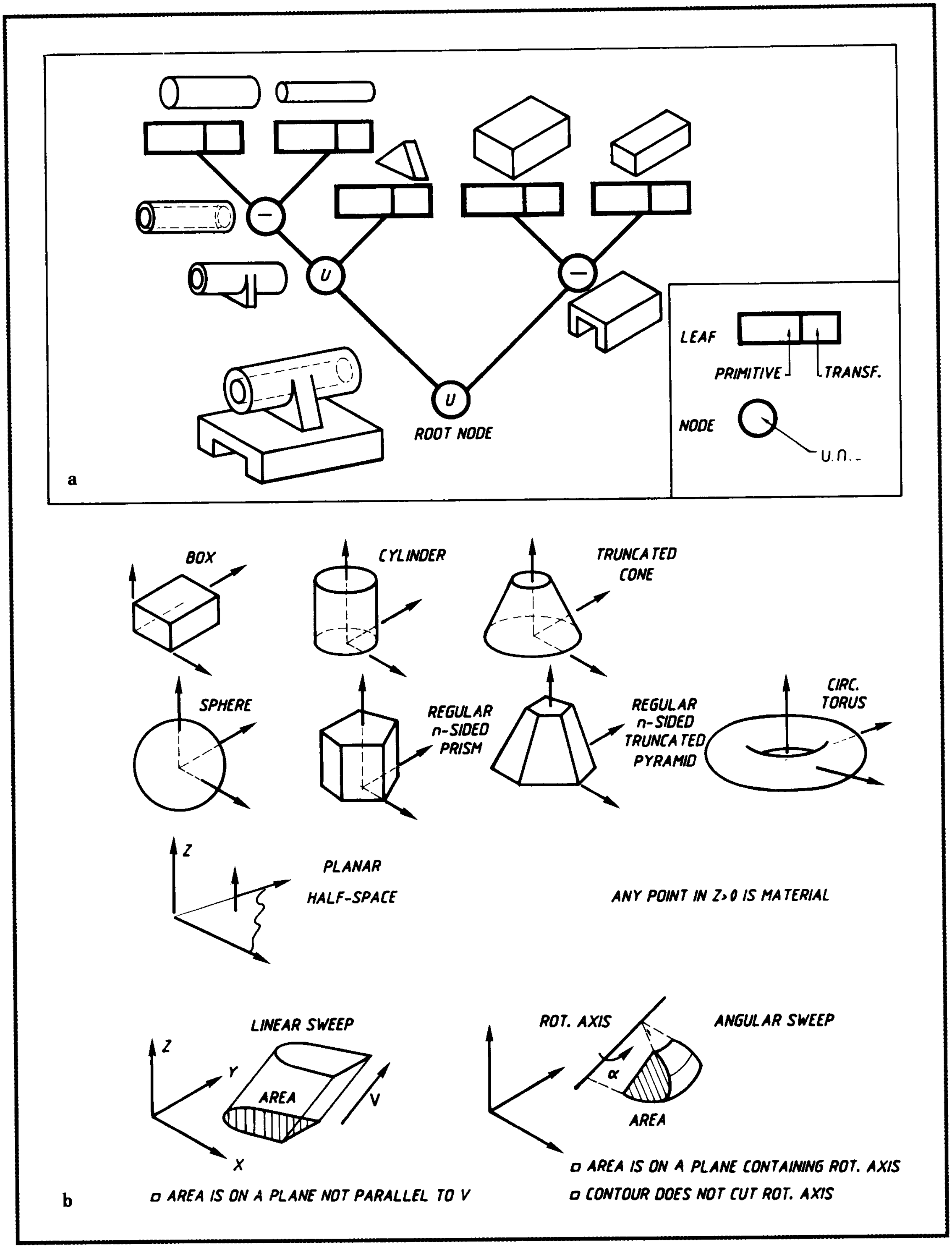


Figure 2. (a) CSG representation of a solid body, (b) basic primitives, the unbounded half-space, and sweep primitives supported in the specification.

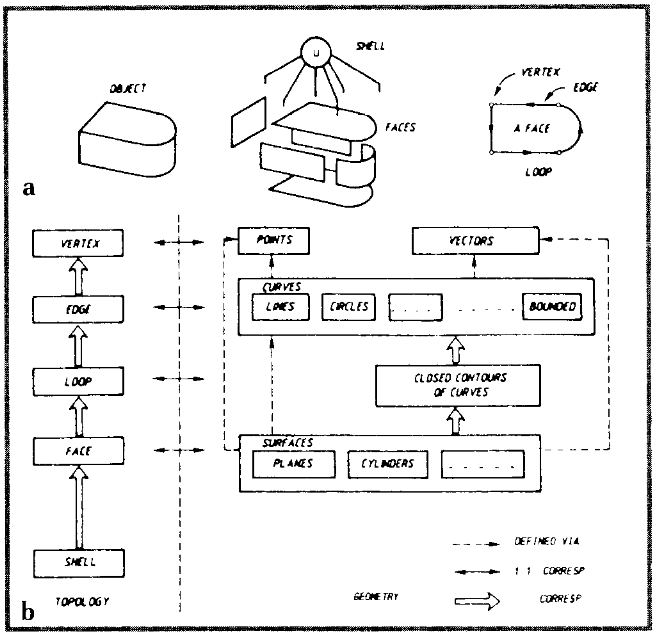


Figure 3. (a) The concept of a boundary representation, (b) the underlying structure to be recorded.

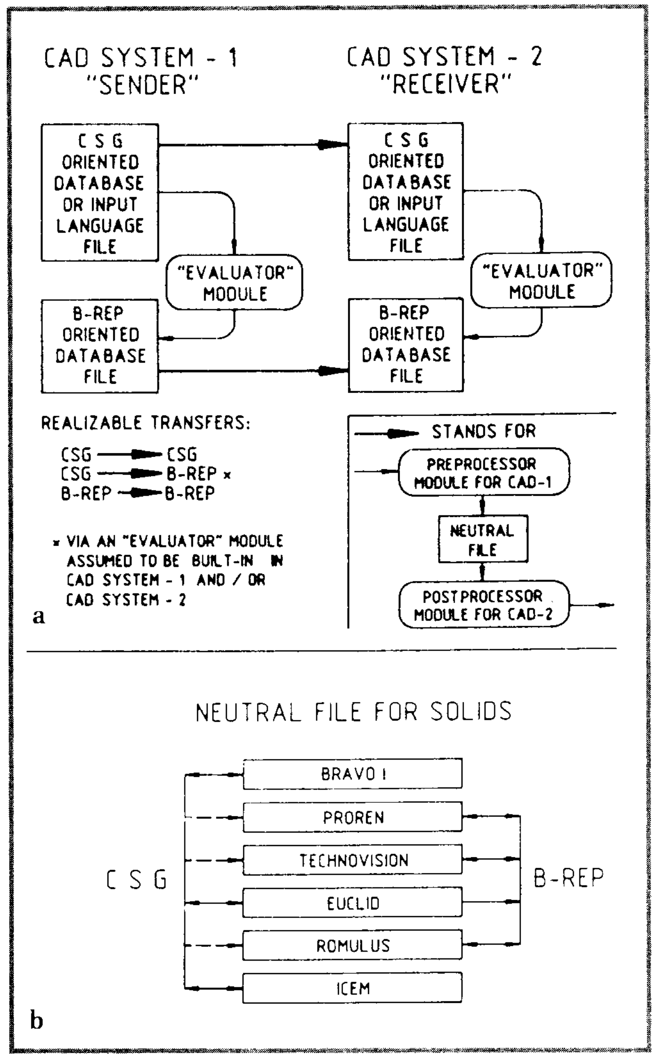


Figure 4. (a) Realizable transfers of solid model data between CAD systems, (b) cycle and intersystem tests within the project's scope.

possible way, in terms of its own entity types.

In the CAD*I specification, three of the most common representations of solids are supported:

- CSG, in which objects are represented by binary tree structures (see Figure 2). The nodes represent combinations of other nodes or leaves via the (regularized) Boolean operators union, intersection, or difference. The leaves are elementary building blocks, instances of primitive solid bodies.
- *B-rep*, in which a solid object is recorded as the collection of its bounding faces. The surfaces corresponding to these faces intersect along curves which, in turn, meet at points (Figure 3).
- *Polyhedral representation*, a particular case of B-rep, in which all surfaces are planar and all curves straight segments. The only geometric information recorded is point coordinates. The topological information builds the correct sequences of these points.

In principle, the specification allows for a B-rep or a polyhedral solid to be a leaf in a CSG tree. In practice, no CAD system until now, to our knowledge, supports such a feature. Moreover, the specification supports "hybrid" models, in which both a CSG construct and its (evaluated) structure in the form of a B-rep exist simultaneously.

Such representations as cell decomposition[14] and octrees[15] are not supported in the specification, since they are not in wide use in commercial systems aimed at mechanical engineering applications. Moreover, none of the partners in the CAD*I project has a CAD system based on these representations.

The realizable exchange of solid geometry information via a neutral description is illustrated in Figure 4. Prototype pre- and postprocessing programs were developed for a number of CAD systems. The programs take advantage of the input and output capabilities of each system. All of the selected systems accept some form of CSG input language. Only two, however, have CSG-oriented internal data structures. One of the systems (Euclid) is hybrid; in other words, it keeps both CSG and B-rep internal representations of models. Although it can send both types, it can receive only CSG-based models via a neutral format.

The consistency of CAD*I neutral files

The diverse neutral files defined by the CAD*I task groups deal with different aspects of CAD and different applications. However, similar entities are represented in a consistent way. This applies, for example, to geom-

Table 1. Entities in the neutral file for solids (Version 2.1) (an asterisk indicates an entity with scope).

```
Grouped  combined  elementary

WORLD*                                         Wireframe entities
  HYBRID_SOLID*                                POINT
  ASSEMBLY*
         COMPONENT                             LINE
         CONSTRUCT*                            CIRCLE
            BOOLEAN
                BOX
                SOLID_SPHERE                   BOUNDED_CURVE
                SOLID_CYLINDER
                TRUNCATED_CONE
                TRUNCATED_PYRAMID
                REGULAR_PRISM
                SOLID_TORUS                    Surface entities
                                               PLANE
                PLANAR_HALFSPACE               CYLINDER

                LINEAR_SWEEP*
                ROTATIONAL_SWEEP*

            COMPOUND_B_REP* (+)
                REGION                         Auxiliary entities
            B_REP*                             INTEGER ┐
                VERTEX                                 │variables
                EDGE                           REAL    ┘
                LOOP
                FACE                           DIRECTION
                SHELL                          VECTOR
                                               PLACEMENT
            POLYHEDRON*
                POLY_LOOP
                POLY_FACE


External references and other CAD information
PART_LIBRARY
RECORD
RECORD_TYPE
ROUT_LIBRARY
ROUTINE*
  FORMAL_PARAMETER
MACRO
```

(+) The COMPOUND_B_REP was tentatively introduced as an experimental entity. It describes a B-rep solid divided in "regions" with different properties as needed for some finite-element applications.

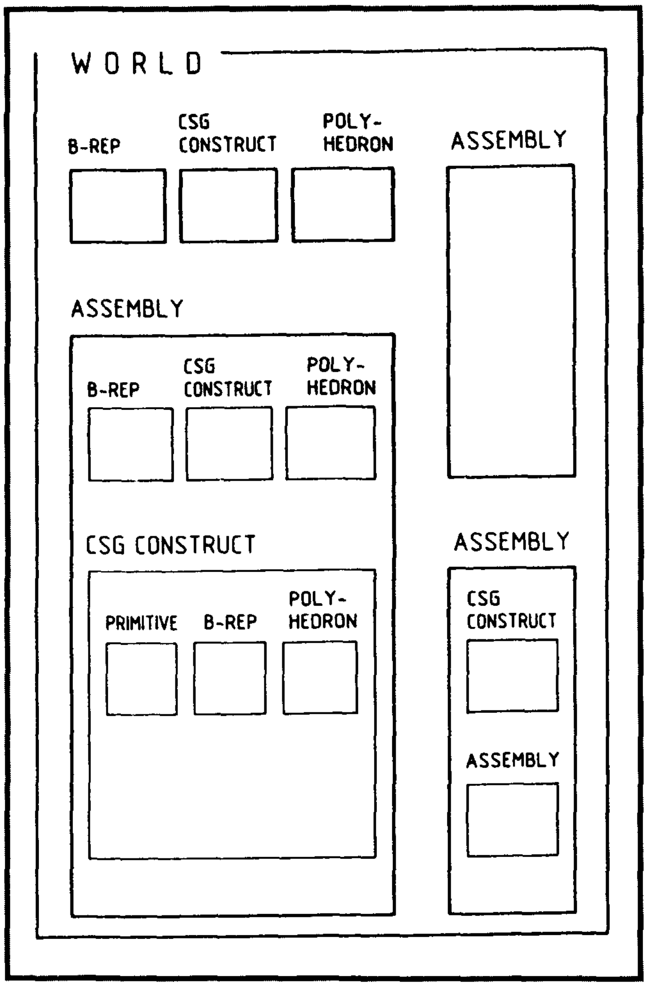


Figure 5. An example illustrating the scope of the reference scheme.

etry defined for a finite-element neutral file, a neutral file for solids, or a neutral file for wireframes. Such auxiliary entities as transformations, general attributes as associated properties, and underlying mechanisms as referencing have identical formats in any neutral file.

Features of the reference scheme for a CAD*I neutral file

As any structured collection of information, the primary components in the neutral format are entities. Describing the exact syntax of how the different entities should actually be written in the neutral file is outside the scope of this article. However, we present a tabular overview of the entities considered in Version 2.1 of the proposal, more or less grouped in classes (see Table 1), as well as some of the underlying guidelines. A more complete description can be found in the CAD*I literature.[11]

Although quite outdated after the release of Versions 3.2 and 3.3, the table is still representative for the purposes of this article.

Scope of entities

The proposal provides for various types of hierarchical relationships among the entities. Probably the most important one is represented by the scope aspect, which represents the "is-known-in" relationship. Although scoping is a well-known concept in modern programming languages, it is less common in data structures.

The scoping principle (sometimes called "block structuring") describes the fact that certain entities can be accessed only in a certain environment. This allows individual entities to be grouped together to some higher level entity. On the next higher level they can be handled only collectively as a single block. If access to details is necessary, provisions are made for a procedural interface to enter the scope of the block and investigate all the constituents. The outermost block is represented by the *world* entity, which may be considered as the complete contents of a CAD database (see Figure 5) and includes in its scope all other entities.

Scoping constitutes an important difference between CAD*I and other approaches (though the concepts of "block" and "subblock" in SET may be regarded as a restricted form of scoping). It enables the transfer of certain behavioral features of a model, besides the geometry, and a formal treatment of operations on blocks of entities that have a meaning in the context of another one.

Another hierarchical structure represents the "is-part-of" relationship and allows the grouping of technical objects. The leaves of this structure are the *component* entities, to which a geometric shape is associated (B-rep, CSG construct, etc.). Components may be grouped to form the *assembly* entities. Assemblies and components can further be grouped to larger assemblies. Each such object can be assigned nongeometric attributes, for example, material properties or user names.

Relations

The definition of some entities often necessitates reference to other entities. Referencing is allowed only to entities defined in the scope of some enclosing entity. Moreover, properties and relationships must belong to the same scope as the entities that they reference. This design criterion enables mapping of each clause in the neutral file onto a procedure and considerably eases the task of postprocessors. To understand some of the referencing capabilities, consider a sphere of radius 2 centered at (3,4,5). Although simplified and not strictly valid from the syntactical point of view, the following descriptions illustrate how the neutral file can accommodate different definition structures, when found in the sending CAD environment:

1. SOLID_SPHERE (#27: POINT (3,4,5), 2);
2. POINT (#26: 3,4,5); SOLID_SPHERE (#27: #26, 2);
3. REAL (#25: 2); POINT (#26: 3,4,5); SOLID_SPHERE (#27: #26, #25); INDEX_ENTRY ('RADIUS', #25; 'CENTER', #26; 'BALL', #27);

In the example above, #25, #26, and #27 are just neutral entity names. The *real* entity in case 3 hints at one of the underlying mechanisms for the description of parametric parts. The parameter *radius* is, in fact, a variable that has the value 2 at transfer time. In the same example, the *index_entry* attribute list relates user-defined names to the entities, so this information is retained as well.

The semantic differences between the three cases in the above example should be noted. While the graphical representation is identical, the behavior following certain operations on a CAD system is different for each case. Such different behavior in a sending system can in fact be conveyed to a receiving system via the CAD*I neutral file.

In case 1, the sphere is a completely self-describing primitive. It does not depend on any other data. In cases 2 and 3, the sphere is centered around a point with neutral name #26. If the point is accessible to operator actions, say shifting to another location, the sphere would follow it as an intended side effect (as would all other entities that also depend on point #26). Wilson named this concept "referential geometry."[16] Similarly, in case 3, a new value could be assigned to the variable *radius* (neutral name #25), which would implicitly change the size of the sphere (and all other entities also depending on entity #25) consistently.

The example illustrates that concentrating on the graphic aspects of CAD data structures (as was done in the initial IGES work), or on the static geometric shape alone, leaves essential parts of CAD models unmapped and is therefore likely to lead to unsatisfactory results.

We point out that even if information of the type in case 3 were present in the sending CAD system database and mapped onto the neutral file without dissipation, its use in the receiving CAD environment can be realized only if the following is true:

- The receiving CAD system supports similar features.
- The postprocessor program performs a thorough translation.
- A set of directives specifying the allowed operations on each entity accompanies the neutral file. These directives are intended for the designer on the receiving side and are transferred via "neutral letters," as outlined later on.

External references

Most references in a neutral file will be internal to entities that are transferred on the same file. A complete design, however, often spreads over various CAD database files as well as shared libraries of frequently used parts. Consequently, external references must be allowed to entities not transferred on the same neutral file.

These are assumed to reside already in the receiving CAD environment, probably as a result of previous transfer events. Furthermore, the scheme supports references to entities in libraries, with the understanding that resolving these part library references is deferred until they are used in the receiving CAD system. The basic mechanism for mapping external references in a neutral file is user-defined names and accompanying directives.

Parametric design

The ability to represent a family of objects similar in their geometric shape or in a functional sense is called "parametric design." The term means different things in different contexts. We have taken up the concept as being ultimately an issue of referencing. From this point of view, the transfer of parametric models is supported in the specifications on three levels.

Library parts

An external reference can be resolved by assigning different actual library parts to the given reference. This allows realization of an assembly with alternative versions of an externally referenced subassembly, as sketched in Figure 6. When the neutral file is read into

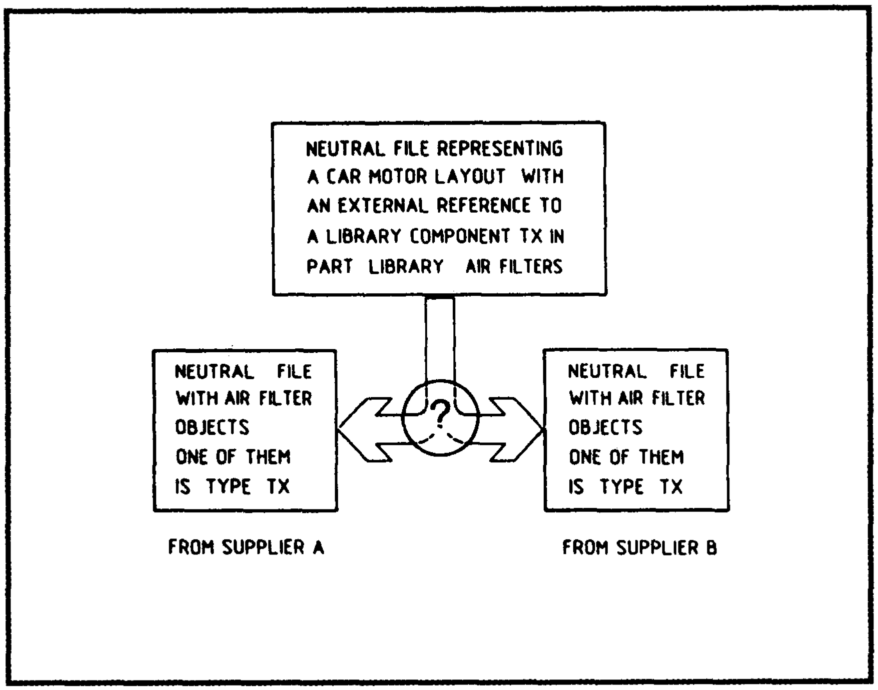


Figure 6. External references and parametric design. Before the postprocessed file is used, one of the two libraries shown must reside in the receiving CAD system.

the CAD system, the external reference should be bound to one of the alternative libraries by the user.

Macros and routines

The way an object is modeled can be determined by invoking a macro sequence of CAD system commands, or perhaps a program made of user-written routines and a collection of CAD system database-access routines. Both methods use variables whose values can be changed, influencing the geometric shape of the construction. These macros and/or routine calls can be mapped onto the neutral file as external references to macro or routine libraries, to the extent they were recorded as such in the sending CAD system's database. The transfer of the contents of these types of libraries is not handled in the neutral file. However, the CAD*I file envelope concept allows the transfer of the source code of such routines and other information in the special "neutral letter" file format, together with the neutral files for solids.

The numerical entity

With this feature, entities of type *integer* and *real* variable are recorded in the neutral file. In contrast with case 3 in the example above, the full definition of these entities consists of the following attributes:

- a (neutral) name
- a mathematical expression involving constants and probably references to other variables
- a value
- a (logical) flag indicating whether the value at transmission time is consistent with the result of the expression when evaluated with the current values of the other variables
- a mode indicating whether the value should be updated at evaluation time when needed (e.g., at display time) or immediately after a parameter value is changed.

A variable might be just a parameter, in which case the expression becomes a constant and the flag indicates consistency. The *index_entry* mechanism associates neutral names to the user names as encountered in the sending CAD system's database. Once transferred into the receiving CAD system, the parameters can be addressed by the designer and their value modified, influencing the geometry. If the receiving system does not have this capability, the values of the variables can still be read from the neutral file and (inconsistent) expression results computed accordingly. At least a "static" model can eventually be recovered after substituting the references with their values. In most places where real (or integer) constants should appear, the specification allows a reference to a corresponding numerical entity instead.

The above three referencing mechanisms add a great deal of flexibility in the transfer of parametric models of moving machine parts, features not usually covered in other transfer protocols.

User records

User records are provided as a way for escaping when necessary from the restrictions imposed by the specifi-

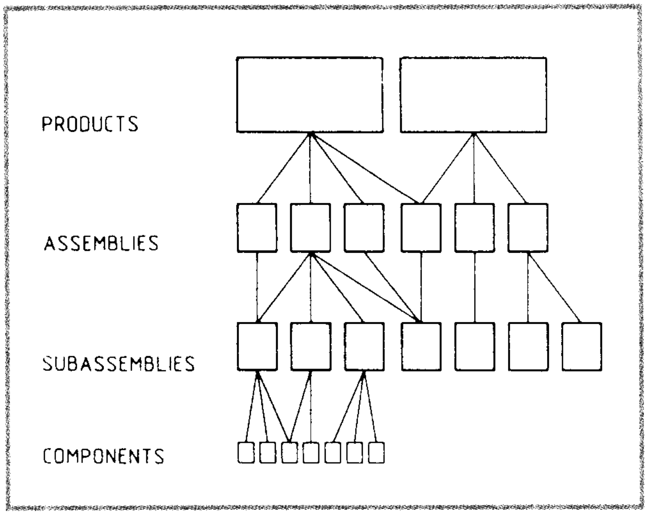


Figure 7. A possible hierarchy in product definition. A full eight-level hierarchy might be *world/product family/product/design version/assembly group/assembly/subassembly/component.*

cation. They are similar to the *declaration section* in PDES and bear some resemblance to the *property entity* in IGES.

User records are not semantically known in the reference scheme. They can be accommodated there for storing information that is to be associated with entities but is to be interpreted by special programs. User records can contain data of types *integer, real, logical,* and *string.*

This mechanism was used to represent the "contour element" concept not covered in Version 2.1 of CAD*I for implementing swept volume primitives in processors for Bravo3 and Technovision.

Levels of scheme implementation (Version 2.1)

The implementation level of particular preprocessors and postprocessors must be able to support certain consistent subsets of the full scheme. The allowable subsets are characterized by four levels:

Lg: geometric modeling capabilities
- 0. Only 2D wireframes
- 1. 3D wireframes
- 2. Surface models
- 3a. Solid models, CSG only
- 3b. Solid models, polyhedral only
- 3c. Solid models, B-rep only
- 4. Solid models of the above three kinds
- 5. Full capabilities including compound B-rep

La: capability for defining assembly structures
- 0. No assembly structure
- 3. Three-level assembly structure in the hierarchy *world/assembly/component*
- 8. Full assembly structure (with unlimited levels of nesting). A possible hierarchy for product description is shown in Figure 7.

Lp: capability for parametric models and macros
- 0. No variables, no macros. All numeric data are constant numbers
- 1a. Variables only, no macros
- 1b. Macros only
- 2. Full parametric capabilities, also *routine*

Lr: capability for external references
- 0. No external references supported
- 1. References to library parts only
- 2. References to external worlds only
- 3. Full external referencing capabilities

The neutral file structure

Rather than give a detailed description, we have chosen to outline some issues relevant in the context of this article. The structure of the neutral language has been formally defined in a High-level Data Specification Language. The syntax, translated from HDSL to Backus-Naur form, is described elsewhere.[11-13]

Several ways of regarding the neutral file had to be discussed for the implementation of the proposal:

- *Physical level:* To exchange sequential files between computers, a protocol was selected for all working groups. For instance, on magnetic tapes such characteristics as density and blocking size are fixed.

- *Metafile level:* According to the CAD*I project conventions, all neutral files consist of card image-format sequential files. This means that they can be considered as a sequence of 80-byte logical records (called cards), where each byte contains the binary representation of the decimal coded alphabet.

All CAD*I metafiles have a common format:

```
CAD*I_FORMAT_BEGIN_19851011 <comment> (1 card)
  one or more neutral files and/or neutral letters
CAD*I_FORMAT_END____19851011 <comment> (1 card)
```

The first card of the metafile is called the header, the last, the trailer. Together they constitute an envelope for the neutral files included. Thus, on a single CAD*I metafile, it is possible to transmit neutral files that were written according to different formats (such as

IGES or VDAFS), as well as computer graphics files and text files, in addition to CAD*I neutral files for solids.

- *Neutral file level:* All neutral files have the following general format:

```
CAD*I_FORMAT_BEGIN_yyyymmdd <comment> (1 card)
  the neutral file contents
CAD*I_FORMAT_END___yyyymmdd <comment> (1 card)
```

where yyyymmdd is the year, month, and day of registration of the neutral format specification at the CAD*I project management board. For Version 2.1 of the neutral file for solids, yyyymmdd = 19860611.

- *Letters:* A useful neutral file that can be used in connection with the one described in this article is the "neutral letter." Basically, letters convey information and directives meant to be read by humans rather than machines. However, they can also be used to transfer the source code of programs. As any other neutral file, a letter starts with a special header card and ends with a corresponding trailer card.

- *Alphabet level:* On this level, a neutral file can be viewed as a continuous stream of symbols from the basic alphabet. The alphabet of the CAD*I neutral file language consists of the set of bytes with ASCII values 32 to 126. Nonstandard and national symbols are handled with special escape sequences.

- *Token level:* On the token level, valid sequences of alphabet characters are defined by the syntax and recognized as keywords, delimiters, etc.

- *Statement level:* The neutral file contents can be regarded as a sequence of "statements" following a determined syntax and having a semantic meaning. The statements are made of tokens in a valid sequence. A statement always terminates with a semicolon. Statements form the "molecules" of the language. As an example, the structures of the *box* and *B-rep* entities are expressed in Backus-Naur form as they appear in the specification (Version 2.1):

```
<box_entity> := BOX (<name>:
     <any(real)>,         (x-dimension)
     <any(real)>,         (y-dimension)
     <any(real)>,         (z-dimension)
     <any(placement)>);   (location and
                           orientation)
```

where <name>, for example, has the form #27 and <any(real)> means that a real constant—for example, 3.465—or a reference to a previously defined *real* entity can be given.

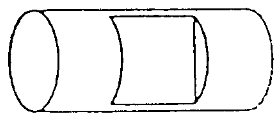
```
<B_rep_entity> := B_REP (<name>: OPEN);
  SCOPE;
    <point_entity_list>
    <direction_entity_list>  (geometry)
    <curve_entity_list>
    <surface_entity_list>

    <vertex_entity_list>
    <edge_entity_list>
    <loop_entity_list>        (topology)
    <face_entity_list>
    <shell_entity_list>

    [<index_entry_property_list>]
  END_SCOPE;
 B_REP (<name>:CLOSE);
```

where <_list> denotes a sequence of similar statements.

Another important issue concerning the physical file is the rule that no entity may be referenced before it is defined on the file. According to this rule, CAD*I postprocessors can, in principle, be coded as single-pass compilers. There are two arguments in favor of the use of backward pointers only; one is related to efficiency and the other to architecture.

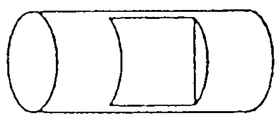
The first argument notes that in a transfer event, as in an interactive design session, the data structure in the receiving CAD system is built as a sequential process, through a large number of steps. After each step, the data structure represents a consistent (yet incomplete) model. Hence "folding out" or "flattening" a complex data structure into a sequence of incremental actions must be performed in any case, either in the sending system while preprocessing to a neutral format or in the receiving system upon postprocessing the neutral file to rebuild the data structure.

If W is the average amount of work required for sequentializing a data structure and K is the average



Start object

a



(b) Pre- and post-processed object

b

```
CAD*I_FORMAT_BEGIN_19851011
CAD*I_FORMAT_BEGIN_19860611
HEADER('B Palstr|m og U Kroszynski','Technical University of Denmark','Norsk Dat
a 530 CX','Sintran','I.F.S.-TECHNOVISION','1986.10.21',,'3c','0','0','0',0,3,2);
WORLD(OPEN);WORLD_HEADER(+1.000E-03,+1.000E+00,+9.218E+04,+9.218E-02);SCOPE;B_RE
P(#1:OPEN);SCOPE;POINT_CONSTANT(#2:+3.000E+01,-1.732E+01,+1.000E+01;#3:+3.000E+0
1,+1.732E+01,+1.000E+01;#4:+7.000E+01,+1.732E+01,+1.000E+01;#5:+7.000E+01,-1.732
E+01,+1.000E+01;#6:+0.000E+00,+0.000E+00,+0.000E+00;#7:+0.000E+00,+2.000E+01,+0.
000E+00;#8:+1.000E+02,+0.000E+00,+0.000E+00;#9:+1.000E+02,-2.000E+01,+0.000E+00;
#10:+3.000E+01,+0.000E+00,+0.000E+00;#11:+7.000E+01,+0.000E+00,+0.000E+00);DIREC
TION(#12:-1.000E+00,+0.000E+00,+0.000E+00;#13:+1.000E+00,+0.000E+00,+0.000E+00;#
14:+0.000E+00,+0.000E+00,>1.000E+00);LINE(#15:#2,#3;#16:#4,#5;#17:#4,#3;#18:#5,#
2);CIRCLE(#19:+2.000E+01,#6,#7,#12;#20:+2.000E+01,#8,#9,#13;#21:+2.000E+01,#10,#
3,#13;#22:+2.000E+01,#11,#5,#12);BOUNDED_CURVE(#23:#21,+0.000E+00,+2.094E+00;#24
:#22,+0.000E+00,+2.094E+00);PLANE(#25:#6,#12;#26:#8,#13;#27:#2,#13;#28:#4,#12;#2
9:#4,#14);CYLINDER(#30:+2.000E+01,#13,#6);VERTEX(#31:#2;#32:#3;#33:#4;#34:#5;#35
:#7;#36:#9);EDGE(#37:#15,#31,#32;#38:#16,#33,#34;#39:#17,#33,#32;#40:#18,#34,#31
;#41:#19,#35,#35;#42:#20,#36,#36;#43:#23,#32,#31;#44:#24,#34,#33);LOOP(#45:(/(:#
41,.T.:)/);#46:(/(:#42,.T.:)/);#47:(/(:#37,.T.:),(:#43,.T.:)/);#48:(/(:#38,.T.:)
,(:#44,.T.:)/);#49:(/(:#38,.F.:),(:#40,.F.:),(:#37,.F.:),(:#39,.T.:)/);#50:(/(:#
41,.F.:)/);#51:(/(:#42,.F.:)/);#52:(/(:#43,.F.:),(:#40,.T.:),(:#44,.F.:),(:#39,.
F.:)/));FACE(#53:#25,(/#45/),.T.;#54:#26,(/#46/),.T.;#55:#27,(/#47/),.T.;#56:#28
,(/#48/),.T.;#57:#29,(/#49/),.T.;#58:#30,(/#50,#51,#52/),.T.);SHELL(#59:(/(:#53:
),(:#54:),(:#55:),(:#56:),(:#57:),(:#58:)/));END_SCOPE;B_REP(#1:CLOSE);END_SCOPE
;WORLD(CLOSE);
CAD*I_FORMAT___END_19860611
CAD*I_FORMAT___END_19851011
```

c

d

```
CAD*I_FORMAT_BEGIN_19851011
CAD*I_FORMAT_BEGIN_19860611
HEADER(
        'B Palstr|m og U Kroszynski',
        'Technical University of Denmark',
        'Norsk Data 530 CX',
        'Sintran',
        'I.F.S.-TECHNOVISION',
        '1986.10.21',,
        '3c',
        '0',
        '0',
        '0',3,3,2);
WORLD(OPEN);
WORLD_HEADER(+1.000E-03,+1.000E+00,+9.218E+04,+9.218E-02);
SCOPE;
B_REP( #1:OPEN);
SCOPE;
POINT_CONSTANT
  (#2:+3.000E+01,-1.732E+01,+1.000E+01;#3:+3.000E+01,+1.732E+01,+1.000E+01;
   #4:+7.000E+01,+1.732E+01,+1.000E+01;#5:+7.000E+01,-1.732E+01,+1.000E+01;
   #6:+0.000E+00,+0.000E+00,+0.000E+00;#7:+0.000E+00,+2.000E+01,+0.000E+00;
   #8:+1.000E+02,+0.000E+00,+0.000E+00;#9:+1.000E+02,-2.000E+01,+0.000E+00;
   #10:+3.000E+01,+0.000E+00,+0.000E+00;#11:+7.000E+01,+0.000E+00,+0.000E+00);
DIRECTION
  (#12:-1.000E+00,+0.000E+00,+0.000E+00;#13:+1.000E+00,+0.000E+00,+0.000E+00;
   #14:+0.000E+00,+0.000E+00,+1.000E+00);
LINE
  (#15: #2, #3; #16: #4, #5; #17: #4, #3; #18: #5, #2);
CIRCLE
  (#19:+2.000E+01, #6, #7,#12; #20:+2.000E+01, #8, #9,#13;
   #21:+2.000E+01,#10, #3,#13; #22:+2.000E+01,#11, #5,#12);
BOUNDED_CURVE
  (#23:#21,+0.000E+00,+2.094E+00; #24:#22,+0.000E+00,+2.094E+00);
PLANE
  (#25: #6,#12; #26: #8,#13; #27: #2,#13;
   #28: #4,#12; #29: #4,#14);
CYLINDER
  (#30:+2.000E+01,#13, #6);
VERTEX
  (#31: #2; #32: #3; #33: #4; #34: #5; #35: #7;
   #36: #9);
EDGE
  (#37:#15,#31,#32; #38:#16,#33,#34; #39:#17,#33,#32; #40:#18,#34,#31;
   #41:#19,#35,#35; #42:#20,#36,#36; #43:#23,#32,#31; #44:#24,#34,#33);
LOOP
  (#45:(/(:#41,.T.:)/);
   #46:(/(:#42,.T.:)/);
   #47:(/(:#37,.T.:),(:#43,.T.:)/);
   #48:(/(:#38,.T.:),(:#44,.T.:)/);
   #49:(/(:#38,.F.:),(:#40,.F.:),(:#37,.F.:),(:#39,.T.:)/);
   #50:(/(:#41,.F.:)/);
   #51:(/(:#42,.F.:)/);
   #52:(/(:#43,.F.:),(:#40,.T.:),(:#44,.F.:),(:#39,.F.:)/));
FACE
  (#53:#25,(/#45/),.T.; #54:#26,(/#46/),.T.;
   #55:#27,(/#47/),.T.; #56:#28,(/#48/),.T.;
   #57:#29,(/#49/),.T.; #58:#30,(/#50,#51,#52/),.T.);
SHELL
  (#59:(/(:#53:),(:#54:),(:#55:),(:#56:),(:#57:),(:#58:)/));
END_SCOPE;
B_REP( #1:CLOSE);
END_SCOPE;
WORLD(CLOSE);
CAD*I_FORMAT___END_19860611
CAD*I_FORMAT___END_19851011
```

Figure 8. (a) The object modeled in the CAD system, (b) the model recovered after postprocessing the neutral file, (c) neutral file in "compressed" mode as produced by the pilot Version 2.1 preprocessor, (d) a "pretty print" of the same neutral file.

number of times any neutral file will be postprocessed (number of necessary passes) to rebuild the data structure, then the benefit from performing the sequentialization in the preprocessor rather than doing it in the postprocessor is (K − 1)W. Thus the advantage of using a strictly sequential file instead of one with "forward pointers" becomes evident.

The second argument for using only backward pointers stems from the long-term design philosophy of the CAD*I project. It is intended to define a programming interface to CAD databases similar to the CAM-I AIS.[7] Building up the data structure representing a model in a transfer event corresponds to a sequence of calls to procedures of the application interface. A neutral format subject to the define-before-use rule has a one-to-one mapping to such calls. Thus, the specification of an application interface will be achieved by converting the neutral file syntax into one compatible with Fortran, C, or any other programming language.

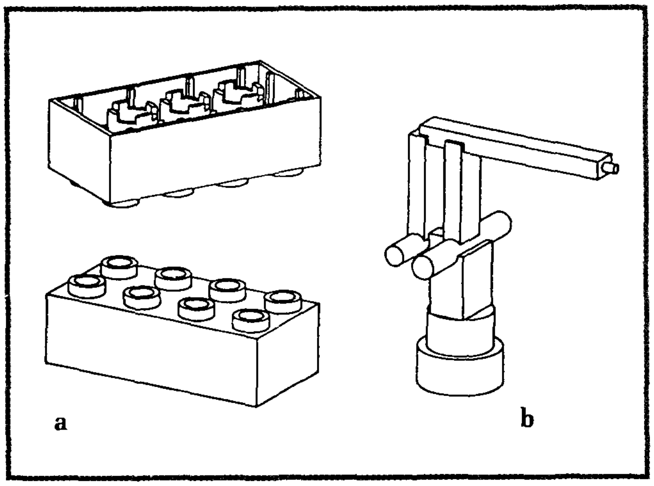


Figure 9. B-rep cycle tests on Technovision models: (a) "Lego brick" and (b) "Hitachi robot."

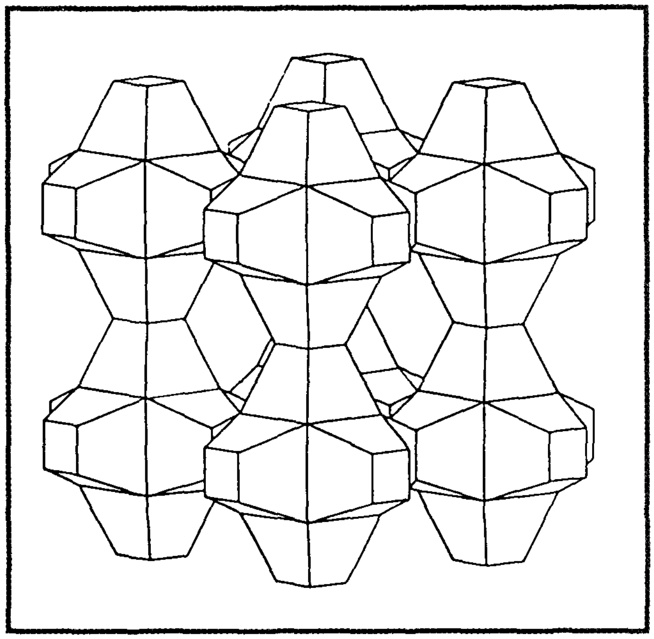


Figure 10. "2*4 module" test object modeled in Proren, as recovered in Technovision.

Examples of solid-model transfer

The first example is a simple object (see Figure 8a) modeled with the Technovision system.[17] Prototype processor programs were coded for mapping B-rep-based descriptions to and from neutral files (at the Control Engineering Institute of the Technical University of Denmark, where cycle tests were carried out). These processors have levels Lg = 3c (and 3b), La = 0, Lp = 0, and Lr = 0.

The preprocessor reads a boundary model from the CAD system database and translates it to neutral format. The postprocessor reads the neutral file back into the system where the model is recovered (see Figure 8b).The neutral file appears as a long sequence of characters packed 80 to the line (see Figure 8c). For readability, Figure 8d gives a "pretty print."

In this cycle test, the only data lost (as compared with the original contents of the native database) were the sizes of the boxes enclosing the surfaces, for which no provision was made in the reference scheme. The size of the box enclosing the entire object (the *world* size) was recorded on the *world_header* statement by the preprocessor, but was not used. These data were not essential for recovering the object. Moreover, the inaccuracy caused by truncation of real numbers after four significant digits was negligible. No further dissipation occurred when performing the cycle again.

Other B-rep models that underwent cycle tests on Technovision are shown in Figure 9.

In mid-1987 the only operational CAD*I postprocessor for B-reps was the one for Technovision. Objects modeled on Isykon's Proren (Figure 10) and Shape Data's Romulus (Figure 11) were transferred to Technovision, showing very good visual and dimensional agreement and perfect model integrity (number of vertices, edges, etc., and their relations). In fact, the design process could be continued in the receiving system.

Because they were prototype implementations, the programs were not optimized (performance, computer resources), nor were further model verifications attempted. All WG2 processors are coded in Fortran 77.

For reference alone, some statistics on the B-rep models above are given in Table 2. The number of cards (NC) in the (compressed mode) neutral files include the headers and trailers. Real numbers in the files occupied 12 bytes each (10 bytes/real for the notched cylinder in Figure 8).

The number of vertices (V), edges (E), loops (L), and faces (F) serve as a measure of the complexity of the model. Following the criterion used by Wilson et al.,[7]

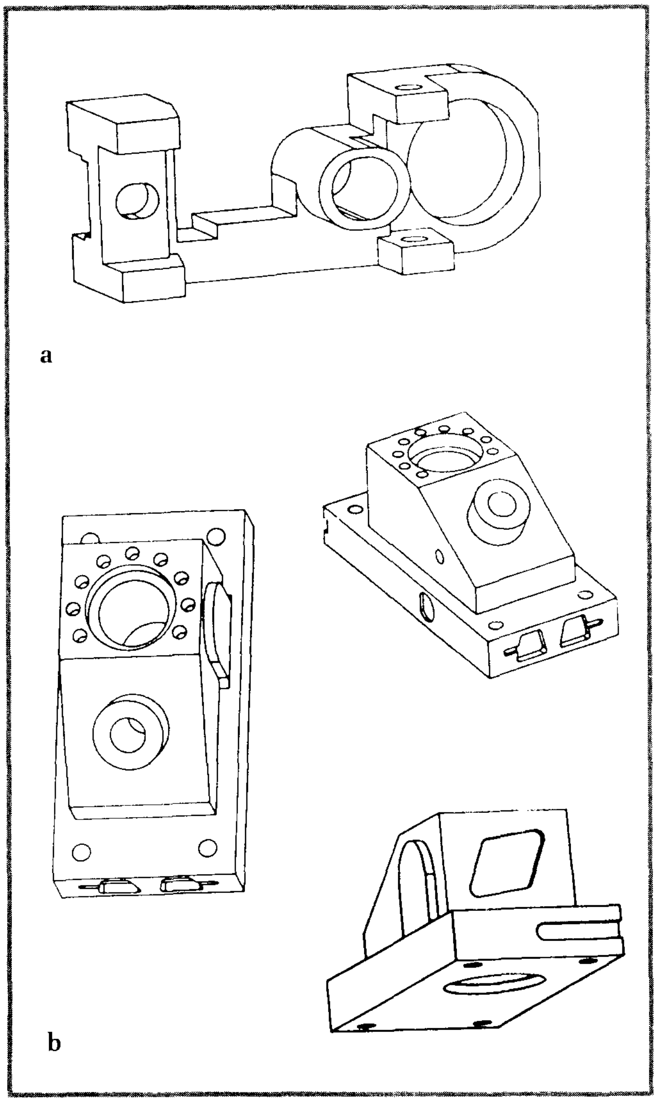


Figure 11. (a) "MBB Gehaeuse" and (b) "ANC101" modeled in Romulus, as recovered in Technovision.

Table 2. Postprocessing of CAD*I B-rep neutral files to Technovision.

Object name	NC	V	E	L	F	S	T	T/S
Notched cylinder	25	6	8	8	6	20	3	.15
HITACHI robot	214	81	121	58	49	251	28	.11
MBB Gehaeuse	271	109	161	70	57	327	35	.11
ANC101	420	146	199	145	95	440	47	.11
LEGO brick	682	259	371	194	154	784	69	.09
2*4 module	918	208	432	216	216	856	77	.09

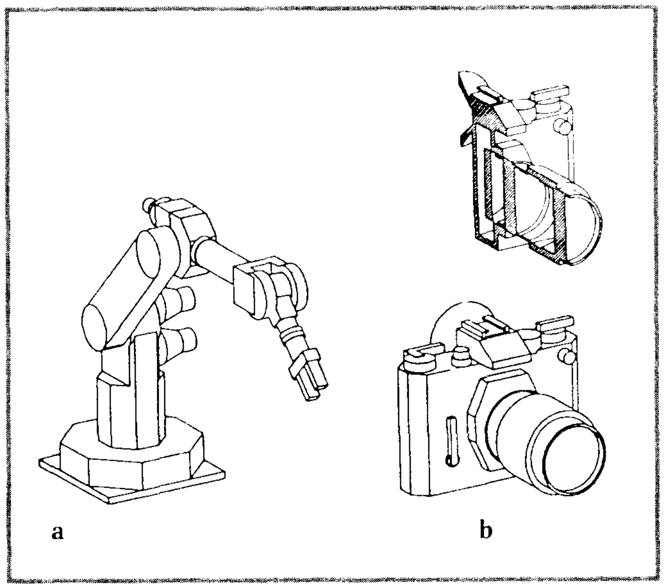


Figure 12. CSG solids transferred to Technovision: (a) "KfK Robot" originated in Euclid, received by Bravo3, and retransmitted to Technovision; (b) "camera" test object from Euclid.

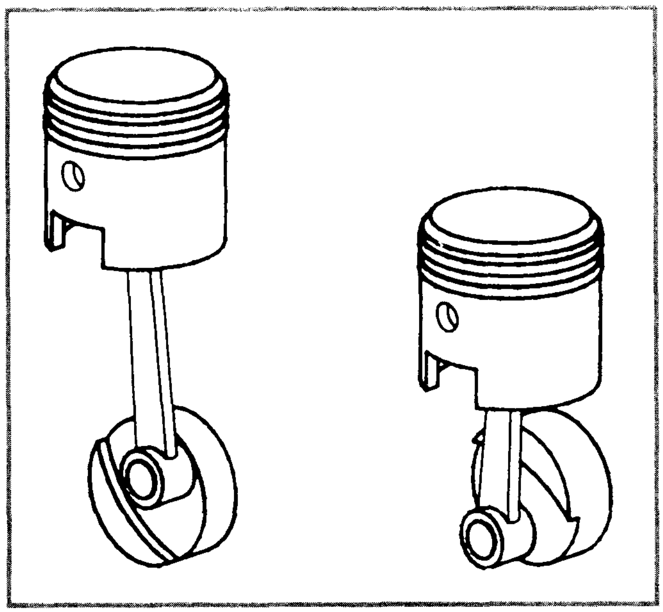


Figure 13. "Parametric piston" generated in the GDS system, after a cycle test, with different values of the shaft angle. The parametric definition is maintained in the transfer via the CAD*I neutral file.

the sum S = V+E+F is noted. For the MBB Gehaeuse the S values are somewhat higher, and for the ANC101 models they are somewhat lower than the respective ones in that article. This is probably because of differences in the degree of detail of the models. According to Table 2, the ANC101 model is 35 percent more "complex" than the MBB. The number is 63 percent when computed from data in Wilson et al.[7]

The processing time T (in seconds) is an overall response time on an ND-530 computer, where the tests were conducted. This time is therefore very difficult to compare with the CPU time reported by Wilson et al.[7] for the BFTODB translator. In Table 2 the processing time includes the scanning and parsing of the neutral file, the actual processing time, and the output to the Technovision database. Although the T/S ratio is roughly constant, the higher values for the simpler objects suggest some constant I/O overhead.

Several CSG-based neutral files corresponding to objects modeled in the Bravo3 and Euclid systems were postprocessed to Technovision. The resulting files are in an APT-like language, supported by this system as off-line input for CSG solids. Figure 12 shows the recovered models.

Figure 13 shows a parametric part modeled in the Geometrical Design System, GDS,[18] a CSG-based research system developed at the Control Engineering Institute. The part is a piston assembly whose shape depends on the shaft angle chosen as parameter. After a cycle test, the parametric structure was recovered in GDS, where the angle value was changed. However, only a static "frozen" model could be recovered in Technovision.

The status of processor development in Working Group 2

Currently operational CAD*I processor programs have been developed at the Control Engineering Institute (IFS) of the Technical University of Denmark, the Kernforschungszentrum Karlsruhe (KfK), the Cranfield Institute of Technology (CIT), and the NEH Consulting Engineers A/S of Denmark (NEH). Table 3 gives an overview.

Because they are prototypes, the processors are incomplete in the sense that not all correspond to the latest version (Version 3.3) of CAD*I. Also, only parts of the specification were implemented, no test data for accuracy checking was included, only preprocessors or only postprocessors were coded in some systems, and no optimization was attempted. Even so, these programs and the tests conducted have provided vital feedback that led to modifications in the early specifications and produced the first intersystem transfers of B-rep solids ever, demonstrating the feasibility of the approach.

Most processors share the same package of low-level routines and the same scanner-parser module. Other software tools were also developed in the framework of WG2, for example, a utility for handling neutral files and letters in a CAD*I metafile.

The relationship to standardization

Although they are national standards, the IGES data format, SET, and VDAFS have serious limitations. In 1984 the International Standards Organization, in its subcommittee ISO/TC184/SC4, resolved that a single

Table 3. Currently operational CAD*I processors (solid models).

Vendor	System	Site	Pre/Post	Type	Lg	La	Lp	Lr
Applicon	BRAVO3	KfK	both	CSG	3a	3	1a	1
Matra Datavision	EUCLID	KfK	both	CSG	3a	3	0	0
- " -	- " -		pre	B_rep	3b	3	0	0
Isykon	PROREN	KfK	both	B_rep	3c	0	0	0
Norsk Data	TECHNOVISION	IFS	both	B_rep	3c	0	0	0
- " -	- " -		post	CSG	3a	0	0	0
Control Data	ICEM	NEH	both	CSG	3a	3	0	1
Shape Data	ROMULUS	CIT	both	B_rep	3c	3	0	2
SDRC	GEOMOD	UKA	both	B_rep	3c	0	0	0
Dassault	CATIA	KfK	pre	B_rep	3b	0	0	0
(Robotic applications and research systems)								
	GBSim	KfK	post	B_rep	3b	0	0	0
	GDS	IFS	both	CSG	3a	3	1a	2
Applicon	BRAVO3 (ext)	KfK	pre	B_rep	3b	0	0	0
BYG	GRASP	IFS	post	B_rep	3b	0	0	0

worldwide Standard for the Exchange of Product model data (STEP) should be developed. It would be based on experience with existing standards, but not on one of them alone. STEP is being developed in ISO/TC184/SC4/WG1, with active participation by France, Great Britain, West Germany, Japan, the Netherlands, Switzerland, and the United States. The US delegation has been assigned to lead the effort. The CAD*I project contributes to this development by actively participating in the national bodies of Great Britain and Germany and in the ISO meetings.

Recently, CAD*I was credited by the inclusion with almost no modification of its neutral geometry description in STEP. Translator programs from CAD*I to a preliminary STEP version were coded and tested.[19] Thus many of the concepts common to both the CAD*I specification and STEP can be tested before the release of the latter as an international standard.

Recent activities

The CAD*I project, originally planned to last for five years, is now close to completion. The latter phase of the project, which was concerned with solids, conducted intensive exchange tests on models of ever-increasing complexity, following upgrades in the capabilities of the processor programs. In particular, after the release of Version 3.3, the underlying geometry for B-reps was significantly enlarged with the inclusion of wireframes. The only curves supported in Version 2.1 were the straight line and the circular arc, and the only surfaces were the plane and the circular cylinder. The tests were intended mainly to identify mapping problems, but they helped to discover errors in some of the CAD systems themselves.

Other aspects of the specification were implemented and tested. Model verification to measure the reliability of model transfer is an important topic.[20] Version 3.3 of CAD*I provides for test data in the form of straight lines to be intersected with the model and coordinates of the expected intersection points. Another criterion is the comparison of such integral model properties as surface area, volume, and location of the center of mass. Extensive tests based on this criterion were carried out with CAD*I as a neutral geometry interface in a robot welding application.[21]

Although a full, consistent implementation of all the wireframe and surface parts of the specification for B-reps is unlikely in the time span of the project, special programs to test the stability and accuracy of certain representations will probably be included in some processors. For instance, CAD*I processors for AutoCAD were coded at IFS to test some aspects of the 2D geometry part of the specification.

Conclusions

The CAD*I specification described in this article is one attempt to standardize the interfaces between the individual subsystems in a CIM environment. Such standardization is requisite in the development of future CIM systems, as existing components are replaced by more effective ones, leading to the application of multivendor systems (the "open system" approach). Standardization and modularity will allow companies to build CIM systems that suit their needs in size and implementation speed. Our presentation of some of the results of the CAD*I project outlines one of the European activities in this context. Overview reports presented at the ESPRIT Technical Week conferences summarize other results of the project.[22-24]

Obviously the integration of computer-assisted methods and tools is having an increased impact on industry, particularly the mechanical industry. Today these computer-assisted systems are implemented on heuristic principles. However, we hope that the concepts developed by the CAD*I project will contribute to the establishment of design rules for CIM systems. We are confident that even small companies, vendors, users, and in the end consumers will benefit from cheaper, more reliable, and better built products. Standardization of modules in automation will contribute to make this happen very soon. ■

Acknowledgments

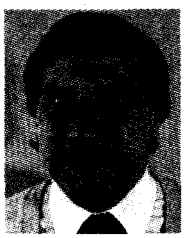
We wish to express our gratitude to the following people at the Kernforschungszentrum Karlsruhe: N. Braendli, U. Gengenbach, F. Katz, M. Mittelstaedt, W. Reichert, and W. Weick. We also thank L. Gungaard, P.B. Hansen, and D. Welner at the Technical University of Denmark, as well as B. Frayssinet, Cisigraph; S.R. Hailstone, Cranfield Institute of Technology; B. Schilli, Universitaet Karlsruhe; and A. Spliid, NEH Consulting Engineers Aps.

References

1. *Digital Representation for Communication of Product Definition Data*, ANSI standard Y14.26M, American National Standards Inst., New York, 1981.
2. VDA Flaechenschnittstelle (VDAFS), Version 1.0, Deutsches Institute fuer Normung (DIN) 66301, Beuth Verlag, Berlin, 1986.
3. *Automatisation industrielle. Representation externe des donnees de definition de produits. Specification du standard d'echange et de transferts (SET)*, Version 85-08, Z68-300, Association Francaise de Normalisation (AFNOR) 85181, Paris, 1985.
4. *IGES Experimental Solids Proposal (ESP)*, US Dept. of Commerce, Washington, D.C., 1986.
5. "CAM-I Geometric Modeling Project Boundary File Design (XBF-2)," Report No. R-81-GM-02.1, CAM-I Inc., Arlington, Texas, 1982.
6. P.R. Wilson: "A Short History of CAD Data Transfer Standards," *CG&A*, Vol. 7, No. 6, June 1987, pp. 64-67.
7. P.R. Wilson et al., "Interfaces for Data Transfer Between Solid Modeling Systems," *CG&A*, Vol. 5, No. 1, Jan. 1985, pp. 41-51.
8. R.A. Carringer, "Integrating Manufacturing Systems Using the Product Definition Data Interface," CASA-SME Tech. Paper MS85-1111, SME, Dearborn, Mich., 1985.
9. W.B. Gruttke, "PDES File Structure Working Draft Version 3," internal notes, PDES Physical File Structure/Formal Language Committee, McDonnell Douglas, St. Louis, Sept. 1985.
10. "Status Report 2" (1986), "Status Report 3" (1987), I. Bey and J. Leuridan, eds., ESPRIT Project 322: CAD Interfaces, CAD*I, Kernforschungszentrum Karlsruhe GmbH, Karlsruhe, West Germany (PFT publications 121 and 132).
11. "Specification of a CAD*I Neutral File for Solids, Version 2.1," ESPRIT Research Reports, CAD*I, Vol. 1, E.G. Schlechtendahl, ed., Springer-Verlag, Heidelberg, West Germany, 1986.
12. "Specification of a CAD*I Neutral File for CAD Geometry, Version 3.2," *ESPRIT Research Reports, CAD*I*, Vol. 1, 2nd ed., E.G. Schlechtendahl, ed., Springer-Verlag, Heidelberg, West Germany, 1987.
13. "Specification of a CAD*I Neutral File for CAD Geometry, Version 3.3," *ESPRIT Research Reports, CAD*I*, Vol. 1, 3rd ed., E.G. Schlechtendahl, ed., Springer-Verlag, Berlin, 1988.
14. I. Carlbom, I. Chakravarty, and D. Vanderschel, "A Hierarchical Data Structure for Representing the Spatial Decomposition of 3-D Objects," *CG&A*, Vol. 5, No. 4, Apr. 1985, pp. 24-31.
15. K. Fujimura et al., "Octree Algorithms for Solid Modeling," in *Computer Graphics Theory and Applications*, T.L. Kunii, ed., Springer-Verlag, Heidelberg, West Germany, 1983, pp. 96-110.
16. P.R. Wilson, "Conic Representations for Shape Description," *CG&A*, Vol. 7, No. 4, Apr. 1987, pp. 23-30.
17. P.B. Hansen et al., "A Description of the Possible Interfaces to Technovision," Report No. CAD*I.WG2.DTH.001/86, internal publication, Technical Univ. of Denmark, Lyngby, Denmark, 1986.
18. U. Kroszynski and A. Sorgen, "Tailoring Intelligent and Versatile

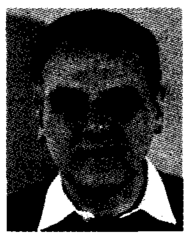
CAD Systems for Different Needs," *Proc. Int'l Conf. Engineering Design*, Vol. 2, Heurista, Zurich, Switzerland, 1983, pp. 649-654.

19. E. Trostmann et al., "CAD Data Exchange Via Neutral Interface," SME Tech. Paper MS88-376, SME, Dearborn, Mich., 1988, pp. 2.95-2.108.
20. R.M. Wilson, "Verification of CAD-Based Data Exchange," CASA-SME Tech. Paper MS85-1113, SME, Dearborn, Mich., 1985.
21. U. Kroszynski, B. Palstroem, and E. Trostmann, "Neutral Interfaces That Work: Application For Robot Welding," *Symp. Information Control Problems in Manufacturing Technology (INCOM)*, IFAC Publications, Oxford, U.K., 1989.
22. I. Bey and J. Leuridan, "ESPRIT Project 322, CAD Interfaces: Interfaces in CAD Systems," *ESPRIT 85 Status Report*, North-Holland, Amsterdam, 1985, pp. 1273-1283.
23. I. Bey and J. Leuridan, "Development of Specifications for Exchange of Product Definition and Analysis Data," *ESPRIT 86, Results and Achievements*, North-Holland, Amsterdam, 1986, pp. 441-451.
24. E.G. Schlechtendahl and U. Gengenbach, "CAD Data Transfer: The Goals and Achievements of Project 322, CAD Interfaces (CAD*I)," *Proc. Ann. ESPRIT Conf.*, North-Holland, Amsterdam, 1987, pp. 1547-1569.



Uri I. Kroszynski is an associate professor at the Control Engineering Institute of the Technical University of Denmark, where he specializes in CAD integration in CIM environments, as well as in all aspects of solid modeling. He has participated as a senior researcher in large national and international projects in both Israel and Denmark. In CAD*I he is a member of Working Group 2. Originally a hydrologist, his interest shifted first to computational hydrodynamics and hydraulics and then to computational geometry.

Kroszynski started his engineering studies at the Universidad de la Republica in Uruguay and continued at Technion in Israel, where he received his BSc, MSc, and DSc degrees in civil engineering in 1969, 1971, and 1974.

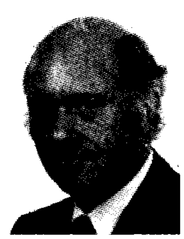


Erik Trostmann is professor of control engineering at the Technical University of Denmark and a project board member of ESPRIT project no. 322, CAD*I. His research interests are in CIM, CAD/CAM, FMS, CNC, and robotics, as well as fluid power control. He has participated in and managed design and development projects in the areas of CNC machine tools, foundry machines, and food production machinery, with responsibility for mechanical design, control system design, and software development. He has also served as advisor and consultant to many private companies, the Danish government, and the ESPRIT commission.

Trostmann received his MSc in mechanical engineering at the Technical University of Denmark in 1954 and his PhD in control theory and engineering at Case Western Reserve University in 1963. He is a member of the Danish Society of Mechanical Engineers, the Danish Academy of Technical Sciences, and Sigma Xi.

Bjarne Palstroem is a member of the research engineering staff of the Control Engineering Institute, where he became associate professor in 1988. His research interests are in programming and control of manufacturing equipment. In the CAD*I team he is the leader of Working Group 4, which deals with network communication.

Palstroem obtained his MSc degree in mechanical engineering at the Technical University of Denmark in 1984. He is a member of the Danish Society of Mechanical Engineers and the Danish Industrial Robot Association.



Ernst G. Schlechtendahl is a senior scientist responsible for CAD and robotics research at the Kernforschungszentrum Karlsruhe GmbH in West Germany. Within CAD*I he leads Working Group 2. He is also a leading member of the ISO committee developing the STEP standard and was a key participant in the development of the GKS standard.

Schlechtendahl received his Diploma in mechanical engineering at the University of Stuttgart in 1962 and his PhD at Karlsruhe. He is member of the KTG (Nuclear Engineering Society) and GI (Computer Science Association).

Kroszynski, Palstroem, and Trostmann can be reached at the Instituttet for Styreteknik (IFS), Danmarks Tekniske Hoejskole, Bygning 424, DK-2800 Lyngby, Danmark. Schlechtendahl can be reached at the Institut fuer Reaktorentwicklung, Kernforschungszentrum Karlsruhe GmbH, Postfach 3640, D-7500 Karlsruhe 1, West Germany.